
Proyecto Final IPDI

— Sistema de monitoreo inteligente
de cajeros usando la red YoloV8n. —

Objetivos del proyecto

- Mostrar la cantidad de personas en una fila de un cajero automático.
- Mostrar el tiempo promedio de uso de un cajero en un tiempo determinado.



Prueba con Yolov7

Los primeros pasos









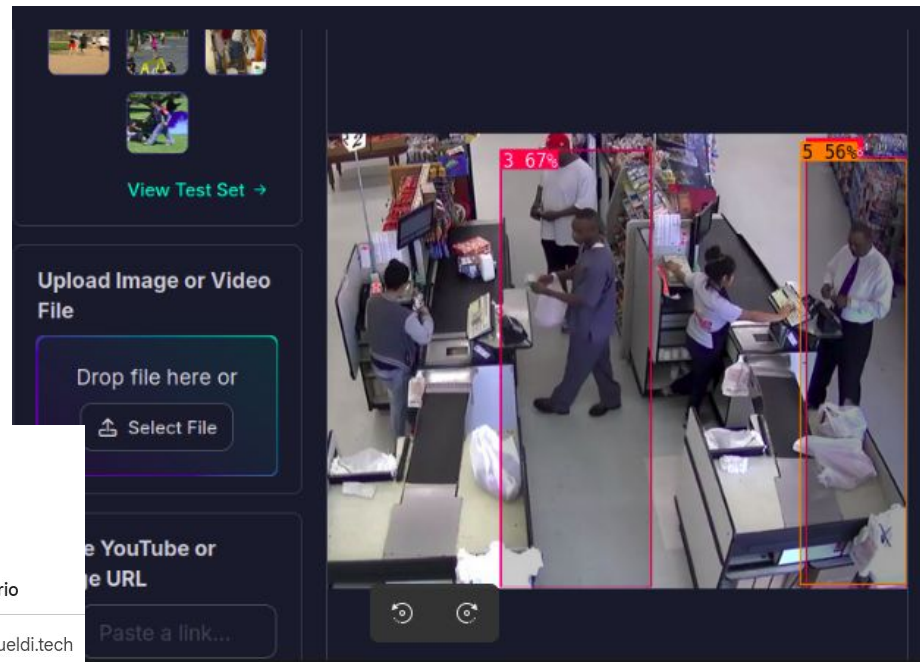
Selección del modelo y pruebas

- Buscamos Videos y Datasets con modelo para probarlo
- Probamos entrenando un modelo propio con colab

... > dataset > train ▾ 👤

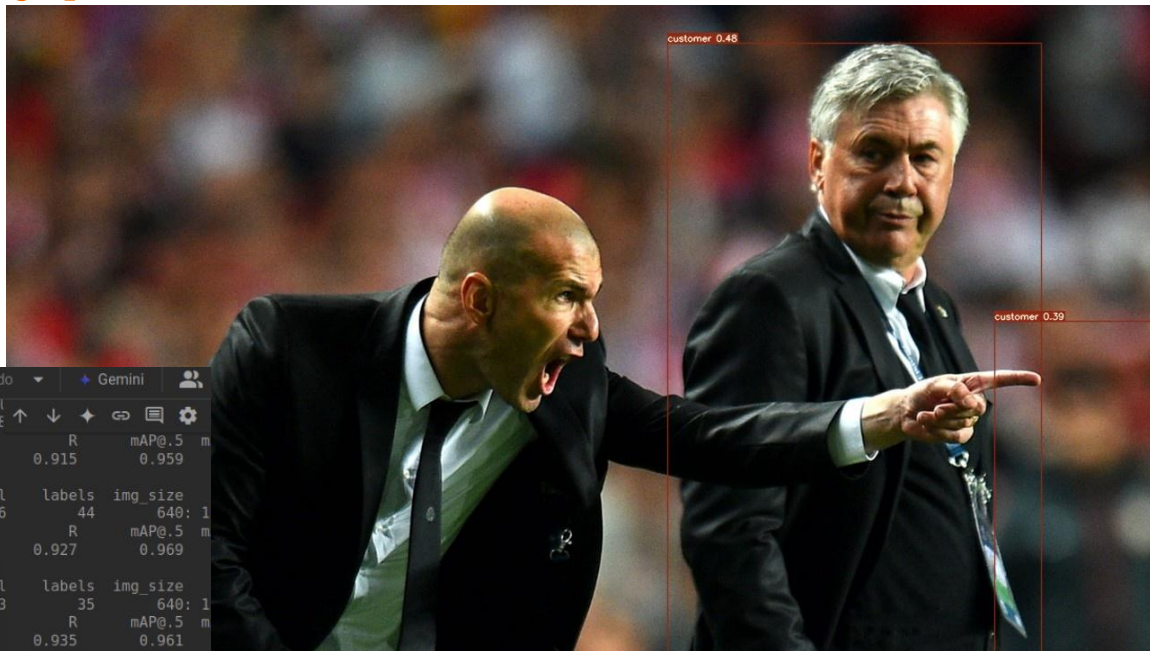
Tipo ▾ Personas ▾ Modificado ▾

Nombre ↑	Propietario
 images	 migueldi.tech
 labels	 migueldi.tech
 labels.cache 👤	 migueldi.tech



Selección del modelo y pruebas

- Los resultados eran insatisfactorios
- Se eligió usar el modelo ya entrenado YoloV8n



Se han guardado todos los cambios

Epoch	gpu_mem	box	obj	cls	total	↑	↓	↔	⚙
20/24	14.2G	0.02314	0.01195	0	0.0350€	P	R	mAP@.5	m
	Class	Images	Labels			0.95	0.915	0.959	
	all	30	248						

Epoch	gpu_mem	box	obj	cls	total	labels	img_size
21/24	14.2G	0.02267	0.01189	0	0.03456	44	640:1
						R	mAP@.5
						0.927	0.969

total	labels	img_size
3453	35	640:1
	R	mAP@.5
	0.935	0.961

No se puede conectar con el backend de GPU

En estos momentos no puedes conectarte a una GPU debido a los límites de uso de Colab. [Más información](#)

Para tener un mayor acceso a GPUs, puedes comprar unidades de computación de Colab con [Pay As You Go](#).

Cerrar Conectarse sin GPU

```
[ ] !find /content/drive/MyDrive/IPDI/Proyecto/prueba2/dataset/train/labels -size 0 -de
```

Desarrollo con YOLOv8n

Mediante el uso el paquete **Ultralytics YOLO**

<https://docs.ultralytics.com/>



Ultralyitcs YOLO

YOLOv8: nuevas funciones y mejoras para aumentar el rendimiento, la flexibilidad y la eficacia.



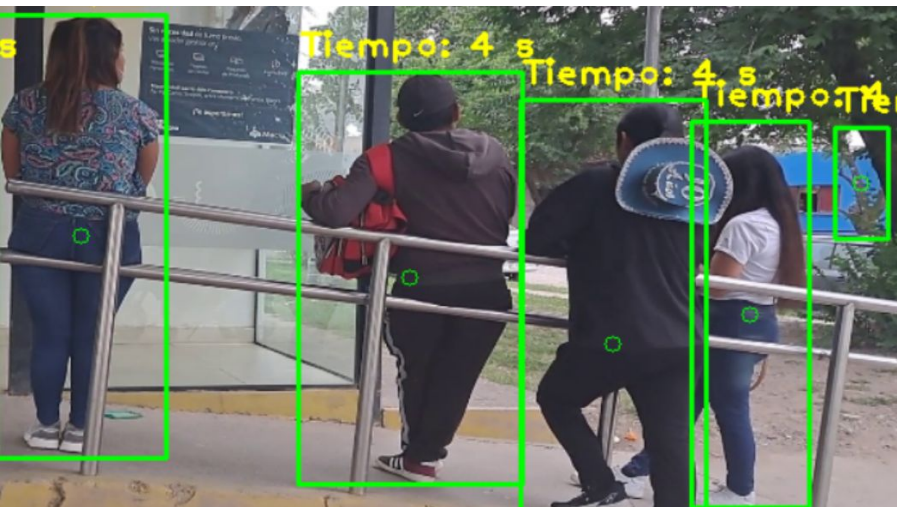
Instalación mediante pip

CLI

```
pip install ultralytics
```

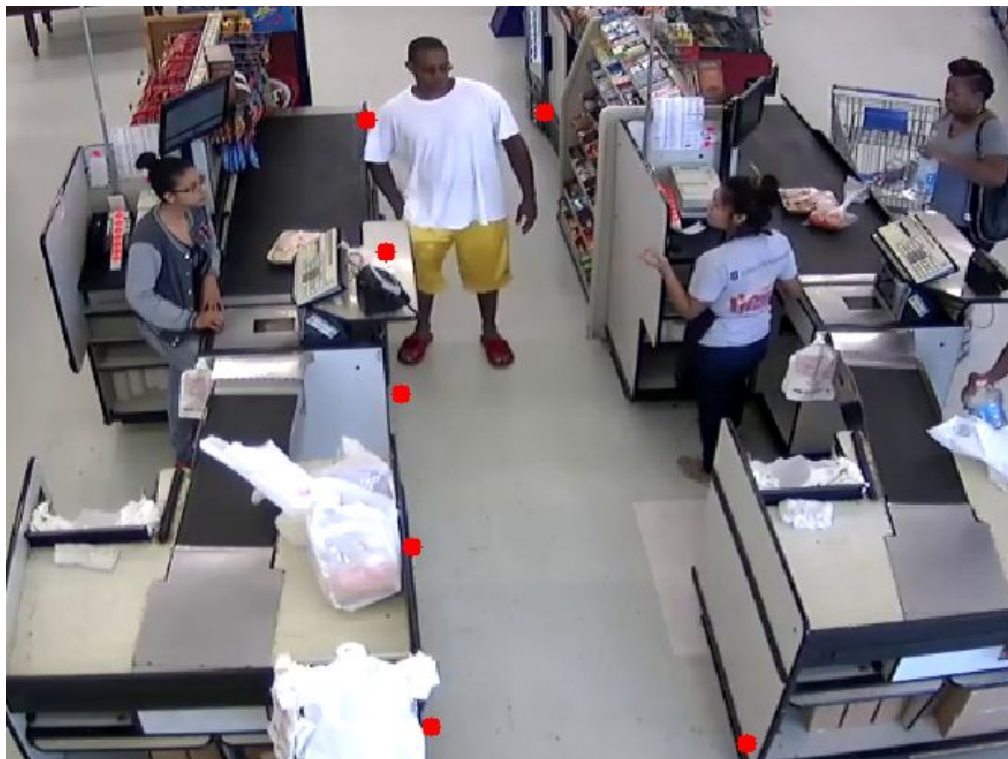


1ra iteración: Filtración de la clase de interés



```
def get_bboxes(results):  
  
    boxes = results[0].boxes  
  
    person_boxes = [  
        box.xyxy[0].cpu().numpy().astype(int)  
        for box in boxes  
        if box.conf[0] >= 0.35 and box.cls[0] == 0  
    ]
```


1ra iteración: Selección de áreas de interés



- `cv2.namedWindow("Frame")`
- `cv2.setMouseCallback("Frame",`
`self.print_coordinates)`
- `self.coordinates.append([`
`x, y])`

2da iteración: Contador del tiempo en area

- Se reiniciaba el tiempo con cada movimiento

```
class Detector:
    def run(self, cap, coords_area1, coords_area2, canvas, update_param):
        detection_timers = {} # Diccionario para rastrear tiempo de detección de personas

        # Tiempo mínimo para calcular promedio (en segundos)
        MIN_TIME_AREA1 = 3 # 3 segundos
        MIN_TIME_AREA2 = 5
        TOLERANCE = 50
        time_spent_by_person = {} # La clave es una combinación de las coordenadas del centro

        # Inicializamos un DataFrame vacío
        #data = pd.DataFrame(columns=["timestamp", "person_id", "entry_time", "exit_time", "du

        #Recorre los fotogramas del vídeo
        while cap.isOpened():
            # Leer fotograma del vídeo
            success, frame = cap.read()
```



2da iteración: Tolerancia de movimiento

```
# Verificaciones de detección y dibujo
if valid_detection(xc, yc, coords_area1):
    center_key = (xc, yc) # Crea una clave única basada en las coordenadas del centro
    #
    for key in time_spent_by_person.keys():
        if abs(key[0] - xc) <= TOLERANCE and abs(key[1] - yc) <= TOLERANCE:
            center_key = key
```

Comparación de
llaves con
TOLERANCE



3ra iteración: Cálculo del tiempo promedio



- Values: [96, 48]
- Cantidad de personas:
2, Promedio: 72.00
segundos

4ta iteración: Interfaz gráfica

- ¿Cómo actualizar los parámetros?

Funciones:

- `__init__()`
- `update_param()`
- `open_video()`
- `play_video()`
- `exit_video()`



```
def update_param(self, avg_time, cant_person):  
    self.tiempo_label.config(text=f"Tiempo Prom.(Area1): {avg_time}")  
    self.personas_label.config(text=f"Cant. Personas (Area2): {cant_person}")
```

Demostración de funcionamiento

Conclusión

¿Qué cambiaríamos?
¿Qué fue lo que mas nos
gusto?



¡Gracias por su atención!